REMARKS

The Examiner is respectfully thanked for the thoughtful consideration provided to this application. Reconsideration of this application is respectfully requested in light of the foregoing amendments and the following remarks.

Each of claims 1-6, 15, 17-23, 34, 36, 39-41, and 43-51 was amended for reasons unrelated to patentability, including at least one of: to explicitly present one or more elements implicit in the claim as originally written when viewed in light of the specification thereby not narrowing the scope of the claim, to detect infringement more easily, to enlarge the scope of infringement, to cover different kinds of infringement (direct, indirect, contributory, induced, and/or importation, etc.), to expedite the issuance of a claim of particular current licensing interest, to target the claim to a party currently interested in licensing certain embodiments, to enlarge the royalty base of the claim, to cover a particular product or person in the marketplace, and/or to target the claim to a particular industry.

Claims 1-52 are now pending in this application. Claims 1, 19, 36, 39, 41, 44, and 51 are the independent claims.

I. The Statutory Subject Matter Rejection

Claim 36 was rejected under 35 U.S.C. 101 as being directed to non-statutory subject matter. Applicant respectfully traverses.

Federal Circuit case law states that "[w]ithout question, software code **alone** qualifies as an invention eligible for patenting under these [35 U.S.C. 101] categories, at least as processes." *Eolas Technologies Inc. v. Microsoft Corp.*, 399 F.3d 1325, 73 USPQ2d 1782 (Fed. Cir. 2005) (*citing In re Alappat*, 33 F.3d 1526 (Fed. Cir. 1994); *AT&T Corp. v. Excel Communications, Inc.*, 172 F.3d 1352 (Fed. Cir. 1999); MPEP § 2106).

Claim 36 recites a "computer program product comprising a computer-readable storage medium having stored thereon a representation of **an industrial automation computer**

program as a markup language version of the industrial automation computer program, the industrial automation computer program adapted for use by a programmable logic controller, the industrial automation computer program created using a graphical programming language".

Thus, claim 36 meets the criteria for patentable subject matter under *Eolas Technologies Inc.*

Accordingly, reconsideration and withdrawal of the rejection of claim 36 is respectfully requested.

II. Claim Construction

On 12 July 2005, the *en banc* Federal Circuit, in *Phillips v. AWH Corp.*, No. 03-1269 (Fed. Cir. 2005), clarified that:

1. "[t]he Patent and Trademark Office ('PTO') determines the scope of claims in patent applications not solely on the basis of the claim language, but upon giving claims their broadest reasonable construction '**in light of the specification as it would be interpreted by one of ordinary skill in the art**'";
2. the words of a claim "are generally given their ordinary and customary meaning";
3. the ordinary and customary meaning of a claim term is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention, i.e., as of the effective filing date of the patent application";
4. "the person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which the disputed term appears, but **in the context of the entire patent**, including the specification";
5. even "the context in which a term is used in the asserted claim can be highly instructive";
6. "the specification may reveal a special definition given to a claim term by the patentee that differs from the meaning it would otherwise possess. In such cases, **the**

inventor's lexicography governs";

7. even "when guidance is not provided in explicit definitional format, **the specification may define claim terms by implication** such that the meaning may be found in or ascertained by a reading of the patent documents";
8. an "invention is construed not only in the light of the claims, but also with reference to the file wrapper or prosecution history in the Patent Office"; and

the "prosecution history... consists of the complete record of the proceedings before the PTO and **includes the prior art cited** during the examination of the patent."

A. Programmable Logic Controller

In the present Application, the customary meaning for the phrase "Programmable Logic Controller" is implicitly defined in the specification and the cited art. That definition must control examination of those claims that recite this phrase.

At least at page 1, the specification of the present Application implicitly defines the term "Programmable Logic Controller (PLC)" by stating that a "PLC may comprise dedicated hardware or, alternatively, be implemented in software on a conventional personal computer, the latter being sometimes referred to as a PC-based PLC."

The International Electrotechnical Commission (IEC) is an organization that prepares and publishes international standards for electrical, electronic and related technologies. IEC standards IEC 1131-1 and IEC 1131-2 are cited as prior art to the present Application in an Information Disclosure Statement filed herewith. IEC standards IEC 1131-1 and IEC 1131-2 provide context in determining a proper meaning for the phrase "programmable logic controller." IEC standard 1131-1 defines the phrase "programmable controller" "(PC)" to mean a:

> *digitally operating electronic system, designed for use in an industrial environment, which uses a programmable memory for the internal storage of user-oriented instructions for implementing specific functions such as logic,*

> *sequencing, timing, counting and arithmetic, to control, through digital or analog inputs and outputs, various types of machines or processes. Both the PC and its associated peripherals are designed so that they can be easily integrated into an industrial control system and easily used in all their intended functions.*

See IEC 1131-1 definition 2.50.

U.S. Patent Number 6,904,471 (Boggs), a patent commonly owned by the assignee of the present Application, is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Boggs provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *[p]rogrammable logic controllers (PLC's) are a relatively recent development in process control technology. As a part of process control, a PLC is used to monitor input signals from a variety of input points (input sensors) which report events and conditions occurring in a controlled process. For example, a PLC can monitor such input conditions as motor speed, temperature, pressure, volumetric flow and the like. A control program is stored in a memory within the PLC to instruct the PLC what actions to take upon encountering particular input signals or conditions. In response to these input signals provided by input sensors, the PLC derives and generates output signals which are transmitted via PLC output points to various output devices, such as actuators and relays, to control the process. For example, the PLC issues output signals to speed up or slow down a conveyer, rotate the arm of a robot, open or close a relay, raise or lower temperature as well as many other possible control functions too numerous to list.*
>
> *The input and output points referred to above are typically associated with input modules and output modules, respectively. Input modules and output modules are collectively referred to as I/O modules herein. Those skilled in the art alternatively refer to such I/O modules as I/O cards or I/O boards. These I/O modules are*

> *typically pluggable into respective slots located on a backplane board in the PLC. The slots are coupled together by a main bus which couples any I/O modules plugged into the slots to a central processing unit (CPU). The CPU itself can be located on a card which is pluggable into a dedicated slot on the backplane of the PLC.*

See, col. 1, lines 16-46.

U.S. Patent Number 6,141,628 (Worth) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Worth provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *PLCs operate by gathering information from various sensor inputs (analog and discrete) and processing the data* ***using Relay Ladder Logic****, a type of computer program based on Hard Wired Relay Logic. As sensor data is gathered and manipulated by the user program, the PLC sends appropriate output signals to control the operation of the equipment to which it is connected. The result is safer, more efficient operation of the monitored or controlled equipment.*

See col. 1, lines 25-32.

U.S. Patent Number 6,819,960 (McKelvey) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. McKelvey provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *[a] programmed logic controller (PLC) executes a series of operations that are performed sequentially and repeatedly. In general, the series of operations includes an input scan, a program scan and an output scan. During the input scan the PLC examines the on or off state of the external inputs and saves these states temporarily in memory (e.g., a file). During the program scan the PLC scans the instruction of the program and uses the input status to determine if an output will be energized. The output results are then saved to memory (e.g., a file). During*

> *the output scan the controller will energize or de-energize the outputs based on the output results stored in memory to control the external devices.*
>
> *A conventional language for programming the stored program is relay ladder logic. Each ladder logic program comprises one or more ladder logic statements, referred to as rungs or instructions. The ladder logic statements define relationships between an output variable and one or more input variables. Input variables are variables that correspond to signals at input terminals and output variables are variables that correspond to signals at output terminals. In relay ladder logic, the input and output signals may be represented graphically as contact symbols and coil symbols arranged in a series of rungs spanning a pair of vertical power rails. A typical ladder logic statement may indicate that a specific output variable is "on" if and only if a first and a second input is "on". The ladder logic program not only manipulates single-bit input and output data representing the state of the sensing and operating devices, but also performs arithmetic operations, timing and counting functions and more complex processing operations.*

See col. 1, lines 26-57.

U.S. Patent Number 6,108,662 (Hoskins) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Hoskins provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *[p]rogrammable controllers are well-known systems for operating industrial equipment, such as assembly lines and machine tools, in accordance with a stored program. In these controllers, a stored program is executed to* ***examine the condition of specific sensing devices on the controlled equipment****, and to energize or de-energize selected operating devices on that equipment contingent upon the status of one or more of the examined sensing devices. The program not only manipulates single-bit input and output data representing the state of the sensing and operating devices, but also performs arithmetic operations, timing and counting functions, and more complex processing operations.*

See col. 1, lines 25-36.

U.S. Patent Number 5,771,374 (Burshtein) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Burshtein provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *[p]rogrammer logic controllers (PLCs) are driven by sequential programs, i.e. all inputs have to be scanned in order to find whether any of the inputs has changed. After all inputs have been scanned, the process takes place in order to deal with the changes according to the ladder diagram program language. Such controllers have the advantage of requiring simple logical programming and are very suitable for alarm and control applications.*

See col. 1, lines 33-40.

U.S. Patent Number 5,970,243 (Klein) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Klein provides context for determining a proper meaning for the phrase "programmable logic controller" by stating that a "PLC is a solid-state device designed to perform logic functions previously accomplished by electromechanical relays. The PLC uses output modules to actuate industrial equipment in response to physical stimuli which the PLC is programmed to recognize through input modules. *See* col. 1, lines 25-30. Klein further recites:

> *[i]ndustrial control programs may be written in relay ladder logic (RLL). RLL referred to herein is a programming language in which input/output signals are written with symbols, such as electrical circuit symbols that conventionally represent relay contacts and relay coils.* ***Control system logic is executed in a repeating sequence of operations consisting of (1) reading all physical inputs, (2) executing the logic once, (3) writing all physical outputs****, and (4) performing any background activity. This sequence is known as one "scan." A RLL control program begins each scan from the top of the ladder diagram. In order to modify*

> *a RLL control program, the user must insert "new" ladder logic rung while marking as "old" any rungs which are replaced, and further marking as "unchanged" any rungs that remain unchanged. To implement the modified relay ladder logic control program, the user executes only the "new" and "unchanged" rungs while retaining the "old" rungs in memory in case the user is forced to return to the previous unmodified control program version. Because relay ladder logic programs fully execute during each scan, the relay ladder logic programming language is referred to as "**stateless**", meaning a relay ladder logic program does not retain any industrial process state information after each scan, unless expressly programmed to do so. As a result, every single input and output is important to each scan in relay ladder logic.*

See col. 1, lines 41-67.

U.S. Patent Number 6,018,797 (Schmidt) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Schmidt provides context for determining a proper meaning for the phrase "programmable logic controller" by stating:

> *[i]ndustrial controllers are special purpose computers used for controlling an industrial process, such as an assembly line, in **real-time** in accordance with a stored program. Under the direction of a stored program, the industrial controller examines a series of inputs reflecting the status of the industrial process and changes a series of outputs controlling the industrial process.*

See col. 1, lines 22-29. Schmidt further recites:

> *[i]n a common relay ladder program, a specialized processor will read each of the rungs (composed of contacts and coils) in sequence to examine contacts and to generate an output via the coil. Each rung is executed **in a consistent order at high speed** until the completion of all rungs. Then there may be a period of refreshing of the I/O data, and the rungs will again be executed. High speed*

> *execution of the rungs provides the appearance that they execute in parallel as would be the case with true relays and as* ***is essential to the fundamental operation of any real-time control program****. A delay or unpredictability in the execution of the control program may have a detrimental effect on the industrial process either causing it to fail or to run erratically. Relay ladder instructions may be executed quickly on simple hardware and provide an intuitive language for control processes.*

See col. 1, line 56 - col. 2, line 4.

Thus, the phrase "programmable logic controller" should be construed as one of ordinary skill in the relevant art would interpret the definition provided in the specification and according to the context provided by patent documents in that same relevant art, such as IEC International Standard 1131-1, IEC International Standard 1131-2, Boggs, Worth, McKelvey, Hoskins, Kreuter, Burshtein, Klein, and Schmidt.

B. "Sequential Function Charts"

In the present Application, the customary meaning for the phrase "sequential function charts" is implicitly defined in the specification and the cited art. That definition must control examination of those claims that recite this phrase.

At least at page 2, the specification of the present Application implicitly defines the term "sequential function charts" by stating that the "symbols available for use via the editor correspond to the particular graphical programming language being used, among which languages are: ladder logic, function block diagrams, **sequential function charts** and flowcharts, and languages if any embodying other formalisms."

The International Electrotechnical Commission (IEC) is an organization that prepares and publishes international standards for electrical, electronic and related technologies. IEC standards IEC 1131-1 is cited as prior art to the present Application in an Information Disclosure

Statement filed herewith. IEC standard IEC 1131-1 provides context in determining a proper meaning for the phrase "sequential function charts." IEC standard 1131-1 defines the phrase "sequential function charts" to mean a "graphical representation of a sequential program consisting of interconnected steps, actions and directed links with transition conditions." *See* definition 2.61.

C. "Function Block Diagrams"

In the present Application, the customary meaning for the phrase "function block diagrams" is implicitly defined in the specification and the cited art. That definition must control examination of those claims that recite this phrase.

At least at page 2, the specification of the present Application implicitly defines the term "function block diagrams" by stating that the "symbols available for use via the editor correspond to the particular graphical programming language being used, among which languages are: ladder logic, **function block diagrams**, sequential function charts and flowcharts, and languages if any embodying other formalisms."

The International Electrotechnical Commission (IEC) is an organization that prepares and publishes international standards for electrical, electronic and related technologies. IEC standards IEC 1131-1 is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. IEC standard IEC 1131-1 provides context in determining a proper meaning for the phrase "function block diagram." IEC standard 1131-1 defines the phrase "function block diagram" to mean a "programming language using function block diagrams for representing the application program for a PC-system" [i.e., programmable logic controller system]. *See* definition 2.30(1).

D. "Ladder Logic"

In the present Application, the customary meaning for the phrase "ladder logic" is

implicitly defined in the specification and the cited art. That definition must control examination of those claims that recite this phrase.

At least at page 2, the specification of the present Application implicitly defines the term "ladder logic" by stating that the "symbols available for use via the editor correspond to the particular graphical programming language being used, among which languages are: **ladder logic**, function block diagrams, sequential function charts and flowcharts, and languages if any embodying other formalisms."

The International Electrotechnical Commission (IEC) is an organization that prepares and publishes international standards for electrical, electronic and related technologies. IEC standards IEC 1131-1 is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. IEC standard IEC 1131-1 provides context in determining a proper meaning for the phrase "ladder logic." IEC standard 1131-1 defines the phrase "ladder diagram language" (a.k.a. "ladder logic" to one of ordinary skill in the art) to mean a "programming language using ladder diagrams for representing the application program for a PC-system". *See* definition 2.30(4).

III. The Anticipation Rejections

Each of claims 1-8, 10-12, 14-26, 28-30, 32-45, and 47-52 was rejected as anticipated under 35 U.S.C. 102(e). In support of the rejection, Dole (U.S. Patent No. 6,634,008) was cited.

Applicant respectfully traverses each of these rejections.

Dole fails to establish a prima facie case of anticipation. *See* MPEP 2131. To anticipate expressly, the "invention must have been known to the art in the detail of the claim; that is, all of the elements and limitations of the claim must be shown in a single prior art reference, arranged as in the claim". *Karsten Mfg. Corp. v. Cleveland Golf Co.*, 242 F.3d 1376, 1383, 58 USPQ2d 1286, 1291 (Fed. Cir. 2001). The single reference must describe the claimed subject matter "with sufficient clarity and detail to establish that the subject matter existed in the prior art and that its

existence was recognized by persons of ordinary skill in the field of the invention". *Crown Operations Int'l, LTD v. Solutia Inc.*, 289 F.3d 1367, 1375, 62 USPQ2d 1917, 1921 (Fed. Cir. 2002). Moreover, the prior art reference must be sufficient to enable one with ordinary skill in the art to practice the claimed invention. *In re Borst*, 345 F.2d 851, 855, 145 USPQ 554, 557 (C.C.P.A. 1965), *cert. denied*, 382 U.S. 973 (1966); *Amgen, Inc. v. Hoechst Marion Roussel, Inc.*, 314 F.3d 1313, 1354, 65 USPQ2d 1385, 1416 (Fed. Cir. 2003) ("A claimed invention cannot be anticipated by a prior art reference if the allegedly anticipatory disclosures cited as prior art are not enabled.") The USPTO "has the initial duty of supplying the factual basis for its rejection." *In re Warner*, 379 F.2d 1011, 154 USPQ 173, 178 (C.C.P.A. 1967).

Specifically, claim **1** recites, yet Dole fails to expressly or inherently teach or suggest a "identifying an internal representation of an industrial automation computer program, **the industrial automation computer program adapted for use by a programmable logic controller**, the internal representation stored in a computer memory, the internal representation created via a graphical programming language".

Claim **19** recites, yet Dole fails to expressly or inherently teach or suggest a "computer readable program code for identifying **an industrial automation computer program adapted for use by a programmable logic controller**, the industrial automation computer program created via a tool and stored in computer memory in the internal representation, the industrial automation computer program created using a graphical programming language".

Claim **36** recites, yet Dole fails to expressly or inherently teach or suggest a "computer program product comprising a computer-readable storage medium having stored thereon a representation of an industrial automation computer program as a markup language version of the industrial automation computer program, **the industrial automation computer program adapted for use by a programmable logic controller**, the industrial automation computer program created using a graphical programming language".

Claim **39** recites, yet Dole fails to expressly or inherently teach or suggest a "industrial automation graphical programming language code, the graphical programming language code comprising an editor adapted to permit the user to create an industrial automation computer program using graphical elements, the industrial automation computer program being stored in memory in an internal representation during execution, **the industrial automation computer program adapted for use by a programmable logic controller**".

Claim **41** recites, yet Dole fails to expressly or inherently teach or suggest a "creating a schema defining a content model for a markup language version of an industrial automation computer program converted from a graphical language version of the industrial automation computer program, **the industrial automation computer program adapted for use by a programmable logic controller**".

Claim **44** recites, yet Dole fails to expressly or inherently teach or suggest a "accessing a markup language version of the industrial automation computer program, the markup language version of the industrial automation computer program converted from a representation created using a graphical programming language, **the industrial automation computer program adapted for use by a programmable logic controller**".

Claim **51** recites, yet Dole fails to expressly or inherently teach or suggest a "receiving data from the plurality of industrial automation program developer systems, the data comprising an industrial automation computer program presented in a markup language version, the markup language version of the industrial automation computer program converted from a representation created using a graphical programming language, **the industrial automation computer program adapted for use by a programmable logic controller**".

Instead, Dole is allegedly directed to a "methodology server contains **design methodologies accessed by the computers**, with the design methodologies **defining steps of designing and testing of integrated circuits**. The computers or methodology server are also in

communication with a compute server. The compute server executes electronic design automation tools as requested". See Abstract.

Accordingly, it is respectfully submitted that the rejection of claims 1, 19, 36, 39, 41, 44, and 51 is unsupported by Dole and should be withdrawn. Also, the rejection of claims 2-8, 10-12, 14-18, 20-26, 28-30, 32-38, 40, 42, 43, 45, 47- 50, and 52, each ultimately depending from one of independent claims 1, 19, 36, 39, 41, 44, or 51, is unsupported by Dole and also should be withdrawn.

IV. The Obviousness Rejections

Claims 9, 13, 27, 31, and 46 were rejected under 35 U.S.C. § 103(a) as being unpatentable over various combinations of Dole and Hoskins (U.S. Patent No. 6,167,406). These rejections are respectfully traversed.

None of the cited references, either alone or in any combination, establish a *prima facie* case of obviousness. "To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure." See MPEP § 2143.

A. Hoskins Teaches Away From Combination

Federal Circuit law indicates that references "that teach away cannot serve to create a *prima facie* case of obviousness." *See*, In re Gurley, 27 F.3d 551, 553, 31 U.S.P.Q.2D (BNA) 1130, 1132 (Fed. Cir. 1994).

Each of independent claims 1, 19, 36, 39, 41, 44, and 51 recite a "markup language" version of an "industrial automation computer program".

Hoskins allegedly recites that "HyperText Markup Language (**HTML**)" (see col. 11, lines 53-54) "**has proven to be inadequate** in the following areas:

- Poor performance;
- Restricted user interface capabilities;
- Can only produce static Web pages;
- Lack of interoperability with existing applications and data; and
- Inability to scale"

(see col. 12, lines 4-12).

Instead of HTML, **Hoskins praises non-mark-up languages** such as Java and Active X for "**Web applications**". *See* col. 11, line 65 – col. 12, line 2; col. 12, lines 20-65.

U.S. Patent Number 6,463,578 (Johnson) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Johnson provides context for determining a proper meaning for the phrase "Java" by stating that:

> ***Java is an object-oriented programming language*** *developed by Sun Microsystems, Mountain View, California. Java is a portable and architecturally neutral language. Java source code is compiled into a machine-independent format that can be run on any machine with a Java runtime system known as the Java Virtual Machine (JVM). The JVM is defined as an imaginary machine that is implemented by emulating a processor through the use of software on a real machine. Accordingly machines running under diverse operating systems, including UNIX, Windows 95, Windows NT, and MacIntosh having a JVM can execute the same Java program.*
>
> *Java Server Page (JSP) technology is a scripting language technology for controlling the content or appearance of Web pages through the use of server-side applications, known as "servlets."* ***Servlets are Java applications*** *that run on a Web*

server to modify Web pages before they are sent to requesting clients. Servlets may be referred to as server-side applets or applications. Similar to the way applets run on a browser and extend a browser's capabilities, servlets run on a Java-enabled Web server and extend the Web server's capabilities. Servlets use classes and methods in the JavaSoft Java Servlet Application Programming Interface (API). The JavaSoft Java Servlet Application Programming Interface (API) is described at http://www.ibm.com/java/servexp/sedocd.html, which is incorporated herein by reference in its entirety. As is known to those skilled in this art, servlets may be local or remote. That is, servlets may reside on a Web server receiving a request from a Web client or may be located on a server remotely located from the Web server receiving a Web client request.

In response to a client request for a Web page, a JSP file referred to in the requested Web page typically is transformed into (or may call) one or more servlets that execute. A JSP file typically contains ***source code in a markup language, such as HyperText Markup Language (HTML) and Extensible Markup Language (XML). This source code typically includes all the information needed to call one or more servlets****. A servlet typically generates an HTML response to a requesting client.*

See col. 1, lines 14-50.

Thus, Johnson distinguishes Java from markup languages. Java is an "object-oriented" language. Johnson indicates that "source code in a markup language" includes information needed to call "servlets" written as "Java applications". Thus, Johnson indicates that Java is not a markup language.

U.S. Patent Number 5,842,020 (Faustini) is cited as prior art to the present Application in an Information Disclosure Statement filed herewith. Faustini provides context for determining a proper meaning for the phrase "Java" by stating that:

[a]nother technology that has ***function and capability similar to JAVA is***

provided by Microsoft and its ***ActiveX*** *technology, to give developers and Web designers the wherewithal to build dynamic content for the Internet and personal computers. ActiveX runs only the so-called Wintel platform (a combination of a version of Windows and an Intel microprocessor), as contrasted with Java which is a compile once, run anywhere language.*

ActiveX includes tools for developing animation, 3-D virtual reality, video and other multimedia content. The tools use Internet standards, work on multiple platforms, and are being supported by over one hundred companies. The group's building blocks are called ActiveX Controls, small, fast components that enable developers to ***embed parts of software in*** *hypertext* ***markup language*** *(HTML)* ***pages****. ActiveX Controls work with a variety of programming languages including Microsoft's Visual C++, Borland's Delphi, Microsoft's Visual Basic programming system and, in the future, Microsoft's development tool for Java, code named "Jakarta." ActiveX Technologies also includes ActiveX Server Framework, allowing developers to create server applications.*

See col. 8, lines 40-53.

Thus, Faustini distinguishes ActiveX from markup languages. Faustini explains that ActiveX has a function and capability similar to Java, an "object-oriented" language. Faustini indicates that, as with Java servlets, ActiveX controls are called by code written in a "markup language". Thus, Faustini indicates that ActiveX is not a markup language.

Thus, Hoskins teaches away from using "a markup language" version of "industrial automation" code. As a result, one of ordinary skill in the art would have no motivation to consider Hoskins for combination with Dole to arrive at the claimed subject matter due to the inadequacies of HTML listed by Hoskins.

Thus, since Hoskins may not be properly combined with Dole, the cited references fail to establish a *prima facie* case of obviousness.

Because no *prima facie* rejection of any independent claim has been presented, no *prima facie* rejection of any dependent claim can be properly asserted. Consequently, reconsideration and withdrawal of these rejections is respectfully requested.

B. Dole is Non-Analogous Art to the Claimed Subject Matter

According to the Federal Circuit, in order to be analogous art for an obviousness rejection, a reference must either be (1) within the field of the inventor's endeavor or (2) reasonably pertinent to the particular problem with which the inventor was involved. *In re Deminski*, 796 F.2d 436, 230 USPQ 313 (Fed. Cir. 1986).

Dole allegedly recites:

> *[a]n environment for designing integrated circuits. Computers include browsers for displaying pages of forms, with the computers in communication with a methodology server and a compute server. The methodology server contains design methodologies accessed by the computers, with the design methodologies defining steps of* ***designing and testing of integrated circuits****. The computers or methodology server are also in communication with a compute server. The compute server executes electronic design automation tools as requested.*

See Abstract.

Thus, Dole relates to designing and testing "integrated circuits".

By contrast, the present Application states that the field of the invention is "graphical programming languages for programmable logic controllers. In particular, the invention concerns a method and system for standardized storage of graphical programming languages. *See* Page 1.

One skilled in the art at the time of the invention would not have found that "designing and testing of integrated circuits" to be in the same field of endeavor as "graphical programming languages for programmable logic controllers".

Likewise, one skilled would not find "designing and testing of integrated circuits" to be "reasonably pertinent to the particular problem with which the inventor was involved" in "standardized storage of graphical programming languages".

Thus, Dole is nonanalogous art to the present Application and is not available as a reference for combination with Hoskins.

Accordingly, Applicant respectfully requests withdrawal of the rejection of each of claims 9, 13, 27, 31, and 46.

C. No Motivation or Suggestion to Combine Dole With Hoskins

According to the Federal Circuit the "mere fact that the prior art may be modified in the manner suggested by the Examiner does not make the modification obvious **unless the prior art suggested the desirability of the modification.**" *In re Fritch*, 972 F.2d 1260, 23 USPQ 2d 1780, 1783-784 (Fed. Cir. 1992) (quoting *In re Fine*, 837 F.2d 1071, 1075, 5 USPQ 2d 1596, 1600 (Fed. Cir. 1988)). In that same case, the Federal Circuit further held that it "is impermissible to use the claimed invention as an instruction manual or 'template' to piece together the teachings of the prior art so that the claimed invention is rendered obvious. This court has previously stated that '[o]ne cannot use hindsight reconstruction to pick and choose among isolated disclosures in the prior art to deprecate the claimed invention.'" *Id.*, 23 USPQ 2d at 1784.

No evidence is presented indicating that one having ordinary skill in the art would look to Dole, a patent relating to "integrated circuit" design and testing, to combine with Hoskins, a patent relating to "controlling one or more aspects of an industrial environment".

Instead, the present Office Action merely recites a virtually unparseable statement that:

> *[i]t would have been obvious for one of ordinary skill in the art at the time the invention was made to apply the circuit synthesis tool and markup conversion as taught by Dole so that ladder logic be also included as part of the graphical language for synthesis and modeling as taught by Hoskins because task and flow control oriented of blocks as taught by Dole can also be applied via a ladder logic so crucial to enable control on the functionality of circuits such as the very useful controller like a PLC as disclosed by Hoskins should this PLC be one of Dole's target design.*

See Page 9.

Even if the reasoning in the argument from the Office Action was understandable and not erroneous, a premise that Applicant respectfully traverses, **no evidence is presented that "the prior art suggested the desirability of the modification"**. Accordingly, the present Office Action fails to present a *prima facie* case of obviousness by the proposed combination of Dole with Hoskins.

Accordingly, withdrawal of the rejections of each of claims 9, 13, 27, 31, and 46 is respectfully requested.

V. Allowable Subject Matter

The following is a statement of reasons for the indication of allowable subject matter: "none of the references of record alone or in combination disclose or suggest the combination of limitations found in the independent claims. Namely,

claims 1-18 are allowable because none of the references of record alone or in combination disclose or suggest 'identifying an internal representation of an industrial automation computer program, the industrial automation computer program adapted for

use by a programmable logic controller, the internal representation stored in a computer memory, the internal representation created via a graphical programming language';

claims 19-35 are allowable because none of the references of record alone or in combination disclose or suggest 'computer readable program code for identifying an industrial automation computer program adapted for use by a programmable logic controller, the industrial automation computer program created via a tool and stored in computer memory in the internal representation, the industrial automation computer program created using a graphical programming language';

claims 36-38 are allowable because none of the references of record alone or in combination disclose or suggest 'computer program product comprising a computer-readable storage medium having stored thereon a representation of an industrial automation computer program as a markup language version of the industrial automation computer program, the industrial automation computer program adapted for use by a programmable logic controller, the industrial automation computer program created using a graphical programming language';

claims 39-40 are allowable because none of the references of record alone or in combination disclose or suggest 'industrial automation graphical programming language code, the graphical programming language code comprising an editor adapted to permit the user to create an industrial automation computer program using graphical elements, the industrial automation computer program being stored in memory in an internal representation during execution, the industrial automation computer program adapted for use by a programmable logic controller';

claims 41-43 are allowable because none of the references of record alone or in combination disclose or suggest 'creating a schema defining a content model for a markup language version of an industrial automation computer program converted from a graphical

language version of the industrial automation computer program, the industrial automation computer program adapted for use by a programmable logic controller';

claims 44-50 are allowable because none of the references of record alone or in combination disclose or suggest 'accessing a markup language version of the industrial automation computer program, the markup language version of the industrial automation computer program converted from a representation created using a graphical programming language, the industrial automation computer program adapted for use by a programmable logic controller'; and

claims 51-52 are allowable because none of the references of record alone or in combination disclose or suggest 'receiving data from the plurality of industrial automation program developer systems, the data comprising an industrial automation computer program presented in a markup language version, the markup language version of the industrial automation computer program converted from a representation created using a graphical programming language, the industrial automation computer program adapted for use by a programmable logic controller".

CONCLUSION

It is respectfully submitted that, in view of the foregoing amendments and remarks, the application as amended is in clear condition for allowance. Reconsideration, withdrawal of all grounds of rejection, and issuance of a Notice of Allowance are earnestly solicited.

The Office is hereby authorized to charge any additional fees or credit any overpayments under 37 C.F.R. §1.16 or §1.17 to Deposit Account No. 50-2504. The Examiner is invited to contact the undersigned at 434-972-9988 to discuss any matter regarding this application.

Respectfully submitted,

Michael Haynes PLC

Michael N. Haynes

Date: 24 August 2005

Michael N. Haynes
Registration No. 40,014

1341 Huntersfield Close
Keswick, VA 22947
Telephone: 434-972-9988
Facsimile: 815-550-8850